WHAT IS CLAIMED IS:

1. A method for verifying user memory validity in an OS (Operating System), comprising:

generating a system call;

declaring certain functions as a safeguard;

verifying a validity of a user buffer using a user buffer address checking function declared as the safeguard;

determining whether the user buffer address checking function is declared as the safeguard by calling an exception processor, if the user buffer address area is not valid;

identifying an identifier of the safeguard by calling a safeguard exception processor, if the user buffer address checking function is identified as a function in the safeguard area;

identifying whether the user buffer address checking function is defined in the system by identifying the safeguard identifier; and

returning an error value to the user process if the user buffer address checking function is defined in the system.

2. The method according to claim 1, further comprising returning a success value to the user process, and at the same time, exiting the declared safeguard function if the user memory is valid.

3. The method according to claim 1, further comprising exiting the user process if the user buffer address checking function is not defined in the system.

4. The method according to claim 1, wherein the step of verifying the validity of a user buffer comprises:

detecting a page within the user buffer; and

determining whether a fault is generated by sequential access(read/write) to the address area of the detected page.

5. The method according to claim 4, wherein the step of verifying the validity of a user buffer further comprises accessing via a Kernel the user buffer address area and verifying the validity of the user buffer.

6. The method according to claim 5, wherein the Kernel processes a fault generated in the Kernel area as a simple error.

7. The method according to claim 5, wherein the Kernel uses a safeguard to process the fault generated in the Kernel area.

8. The method according to claim 1, wherein the step of verifying the validity of a user buffer is performed without using a MMU (Memory Management Unit) Table.

9. The method according to claim 1, wherein each declared safeguard area has a unique identifier.

10. A method for verifying user memory validity in an operating system (OS), comprising:

performing a system call;

declaring a validity checking function as a safeguard function;

identifying whether the validity checking function is declared as the safeguard function by calling an exception processor if the user memory area is not valid;

calling a safeguard exception processor; and

identifying an identifier of the safeguard exception processor if the validity checking function is in the safeguard function area;

recognizing via the safeguard exception processor that a subject of the process is the validity checking function and identifying whether validity checking the function is defined in the system through the identifier of the safeguard function; and

processing the validity checking function as defined in the system which performs the process of the function, if the validity checking function is defined in the system.

11. The method according to claim 1, further comprising verifying a validity of a user memory area using the validity checking function, wherein the method further

comprises returning a success value to the user process, and at the same time, exiting the declared safeguard function, if the user memory area is valid.

12. The method according to claim 11, further comprising exiting the user process when the validity checking function is not defined in the system.

13. The method according to claim 12, further comprising accessing via a Kernel a user memory area and verifying a validity of the user buffer.

14. The method according to claim 13, wherein the Kernel treats a fault generated in the Kernel area as a simple error.

15. The method according to claim 13, wherein the Kernel uses the safeguard function in order to process the fault generated in the Kernel area.

16. The method according to claim 10, wherein the validity verifying step is executed without using a MMU (Memory Management Unit) Table.

17. The method according to claim 10, wherein the declared safeguard area include a unique identifier.

18. A computer-readable medium having stored thereon a sequence of instructions which, when executed by a processor, cause the processor to at least perform a method comprising:

generating a system call;

declaring a safeguard function;

verifying validity of a user buffer using a user buffer address checking function declared as the safeguard function;

determining whether the user buffer address checking function is declared as the safeguard function by calling an exception processor, if the user buffer address area is not valid;

establishing an identifier of the safeguard function by calling a safeguard exception processor, if the user buffer address checking function is identified as a function in the safeguard area;

confirming whether the user buffer address checking function is defined in the system by identifying the safeguard function identifier; and

returning an error value to the user process if the user buffer address checking function is defined in the system.

19. The computer-readable medium of claim 18 wherein the sequence of instructions further causes the processor to perform the step of returning a success value to

the user process, and at the same time, exiting the declared safeguard function if the user memory is valid.

20. The computer-readable medium of claim 18 wherein the sequence of instructions further causes the processor to perform the step of exiting the user process if the user buffer address checking function is not defined in the system.

21. The computer-readable medium of claim 18 wherein the sequence of instructions further causes the processor to perform the steps of:

detecting a page within the user buffer; and

determining whether a fault is generated by performing at least one read/write function to the address area of the detected page.

22. The computer-readable medium of claim 21 wherein the step of verifying validity comprises accessing via a Kernel the user buffer address area and verifying the validity of the user buffer.

23. The computer-readable medium of claim 22 wherein the step of verifying validity further comprises processing via the Kernel a fault generated in the Kernel area as a simple error.

24. The computer-readable medium of claim 22 wherein the step of verifying validity further comprises using within the Kernel a safeguard function to process the fault generated in the kernel area.

25. The computer-readable medium of claim 18 wherein the step of verifying the validity of a user buffer is performed without using a memory management table.

26. The computer-readable medium of claim 18 wherein each respective safeguard function has a unique identifier.